



RAPPORT

DE LA PROGRAMMATION AVANCEE

&&



STRUCTURES DE DONNEES

Thème: -Jeux tic tac toe

*Filière :* 2ème Année préparatoire (G22)

Réalisé par : -Elmouden Badre

## Année universitaire : 2023-2024

*REMERCIEMENTS*

J' adresse ensuite ma sincères remerciements à Monsieur Nadif et Madame Elbelghiti, qui a généreusement consacré leur temps et leurs effort pour m' accompagner dans cette fromation. Leur soutien indéfectible et leur expertise précieuse ont été déterminants pour la réussite de ce travail.

Tout d'abord, j'exprime ma profonde gratitude à Dieu tout puissant pour m'avoir donné la volonté et la persévérance nécessaires pour mener à bien ce projet avec une ambition claire et ciblée

Avant de présenter ce travail, il est important de commencer

ce rapport par des remerciements à ceux qui nous ont soutenus tout au long de ce mini-projet.

Merci à tous !

Sommaire :

Table de matière

Introduction 4

Chapitre I : Conception et modélisation

1.architecture du programme……………………………………………………………………………………………..6

2.[La méthode UML 6](#_TOC_250002)

3.[Diagrammes de classes………………………………………………………………………………………………….6](#_TOC_250001)-7

4.Diagrammes de sequences……………………………………………………………………………………………….7

5.Decomposition modulaire………………………………………………………………………………………………8

6-Algorithmes Clés……………………………………………………………………………………………………………..8

7-choix de conception………………………………………………………………………………………………………..9

Chapitre II : Réalisation de l’application

1. Les outils de développement 11
   1. Logiciels et outils utilisés……………………………………………………………………………………..12-13
   2. [Langages utilisés………………………………………………………………………………………………….14-15](#_TOC_250000)
   3. Les structures utilisée………………………………………………………………………………………………16
   4. Les fonctions utilisées…………………………………………………………………………………………17-19
      1. Fonction main………………………………………………………………………………………………….20-22
   5. Resultat……………………………………………………………………………………………..23

3- Conclusion ..4-Sources

*Introduction :*

Le jeu Tic Tac Toe, également connu sous le nom de Morpion, est un jeu de réflexion classique et simple qui oppose deux joueurs. Dans le cadre de notre mini-projet, nous avons développé une version numérique de ce jeu en utilisant le langage de programmation C. Ce rapport présente une analyse détaillée de la conception et de l'implémentation de notre code.

L'objectif de ce projet est de fournir une version informatique du Tic Tac Toe, avec une interface utilisateur intuitive, des règles de jeu claires et une logique de jeu robuste. En programmant ce jeu, nous visons à offrir une expérience ludique tout en mettant en pratique nos compétences en programmation et en résolution de problèmes.

Ce projet se concentre sur plusieurs aspects clés : la représentation graphique de la grille de jeu, la gestion des tours des joueurs, la vérification des conditions de victoire ou de match nul, et l'optimisation de l'algorithme pour garantir un gameplay fluide et sans erreurs.

Dans cette introduction, nous allons examiner les concepts fondamentaux du jeu Tic Tac Toe, présenter l'architecture générale de notre programme et expliquer les choix de conception que nous avons faits pour créer une version interactive et engageante de ce jeu classique

Chapitre I :

Conception et modélisation

## Architecture du programme

Le programme Tic Tac Toe est conçu en utilisant une architecture modulaire qui sépare les différentes responsabilités du jeu en plusieurs composants distincts. Les principaux composants sont : la gestion de la grille de jeu via une liste chaînée, les joueurs et leurs interactions, ainsi que les algorithmes de vérification des conditions de victoire.

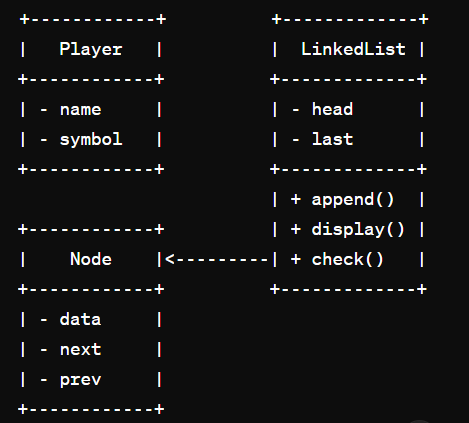
1. La méthode UML

L'UML (Unified Modeling Language) ou Langage de Modélisation Unifié est une méthode de modélisation visuelle utilisée pour représenter les structures, les comportements et les interactions d'un système logiciel.

Les diagrammes UML offrent une vue globale du système, permettant aux concepteurs et aux développeurs de mieux comprendre, communiquer et documenter le système logiciel. Ils peuvent être utilisés à différentes étapes du processus de développement, de la conception initiale à la maintenance et l'évolution du système.

## Diagrammes de Classes

Le diagramme de classes montre les principales classes et leurs relations :

* Node : Représente un nœud de la liste chaînée.
* LinkedList : Représente la grille de jeu sous forme de liste chaînée.
* Player : Représente un joueur avec un nom et un symbole.

*Dans le cadre de l'implémentation du jeu Tic Tac Toe avec une liste chaînée, le diagramme de classe propose une représentation structurée des différentes entités et de leurs interactions.*

### La classe **Node** symbolise les éléments de la liste chaînée, avec des attributs pour stocker les données du tableau (représentant les symboles X ou O) et des références vers les nœuds précédents et suivants. Cette classe encapsule les fonctionnalités de gestion des nœuds,

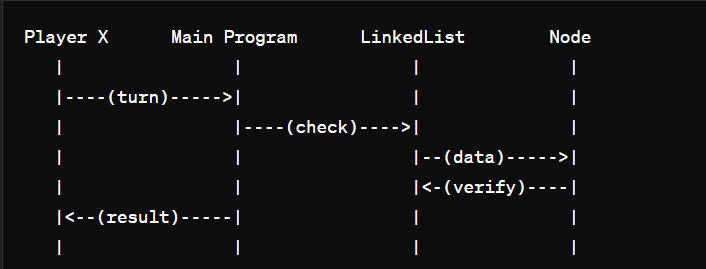
### telles que la création et l'accès aux données.

### La classe **LinkedList** représente la structure globale de la liste chaînée, avec des attributs pour la tête et la dernière case de la liste. Elle offre des méthodes pour manipuler la liste, comme l'ajout de nouveaux nœuds et l'affichage du tableau de jeu. Cette classe facilite la gestion de la structure de données sous-jacente du jeu.

### Enfin, la classe **Player** représente les joueurs du jeu, avec des attributs pour stocker leurs noms et symboles respectifs. Cette classe peut également inclure des méthodes pour initialiser les joueurs avec des valeurs spécifiques et éventuellement gérer leurs actions dans le jeu.

*4-Diagramme de sequences:*

Le diagramme de séquence montre l'interaction entre les objets principaux lors du déroulement d'un tour de jeu*:*



- Le jeu démarre et affiche le menu.

 Le joueur choisit l'option de jouer.

 Le jeu affiche la grille de jeu vide.

 Le joueur X est invité à saisir une position.

 Le joueur X saisit une position.

 Le jeu vérifie si la position est valide et met à jour la grille.

 Le jeu vérifie s'il y a un gagnant ou un match nul.

 Si le jeu n'est pas terminé, le joueur O est invité à saisir une position.

 Les étapes 5 à 8 sont répétées jusqu'à ce qu'il y ait un gagnant ou un match nul.

 Le jeu affiche le résultat de la partie et retourne au menu principal.

*4-Démenstration modulaire:*

 **Gestion de la Grille de Jeu (LinkedList) :**

* createNode(data) : Crée un nœud avec la donnée spécifiée.
* createLinkedList() : Initialise une nouvelle liste chaînée.
* append(data) : Ajoute un nouveau nœud à la liste chaînée.
* displayBoard() : Affiche la grille de jeu.
* checkWinner(symbol) : Vérifie si le joueur avec le symbole spécifié a gagné.

 **Joueur (Player) :**

* Contient le nom et le symbole du joueur

5- Algorithmes Clés:

**Algorithme de Vérification de la Victoire :** L'algorithme de vérification de la victoire parcourt la grille pour vérifier si un joueur a réussi à aligner trois de ses symboles horizontalement, verticalement ou en diagonale

La fonction checkWinner dans le code fourni est chargée de vérifier s'il y a un gagnant dans le jeu Tic Tac Toe en examinant l'état actuel de la grille de jeu représentée par une liste chaînée. Elle prend deux paramètres en entrée : la liste chaînée représentant la grille de jeu et le symbole du joueur pour lequel on veut vérifier s'il a gagné.

La fonction vérifie les différentes configurations gagnantes possibles dans le jeu Tic Tac Toe :

1. Elle parcourt les lignes horizontales, verticales et diagonales pour rechercher si trois symboles identiques apparaissent consécutivement. Si tel est le cas, elle renvoie 1 pour indiquer qu'il y a un gagnant.
2. Elle commence par vérifier les lignes horizontales. Si elle trouve trois symboles identiques dans une ligne horizontale, elle renvoie 1.
3. Ensuite, elle vérifie les colonnes verticales. Si elle trouve trois symboles identiques dans une colonne verticale, elle renvoie également 1.
4. Enfin, elle examine les diagonales principales et secondaires. Si elle trouve trois symboles identiques dans l'une ou l'autre de ces diagonales, elle renvoie également 1.

Si aucun des scénarios gagnants n'est rencontré, la fonction renvoie 0, indiquant qu'il n'y a pas encore de gagnant dans la grille de jeu.

6-Choix de Conception:

**Justification des Choix de Conception :**

* **Utilisation d'une Liste Chaînée :** La liste chaînée est utilisée pour représenter la grille de jeu afin de faciliter l'ajout et la suppression dynamique de nœuds, bien que dans ce cas précis, la grille soit statique. Cela permet également de démontrer la manipulation de structures de données dynamiques.
* **Séparation des Responsabilités :** La séparation des responsabilités entre les différents modules (gestion de la grille, logique de jeu, interactions des joueurs) permet de rendre le code plus modulaire et maintenable.

**Alternatives Considérées :**

* **Tableau 2D :** Une alternative simple aurait été d'utiliser un tableau 2D pour représenter la grille de jeu. Cependant, l'utilisation d'une liste chaînée permet de mettre en pratique des concepts de structures de données plus complexes, ce qui est bénéfique pour l'apprentissage.

Chapitre II :

Réalisation de l’application

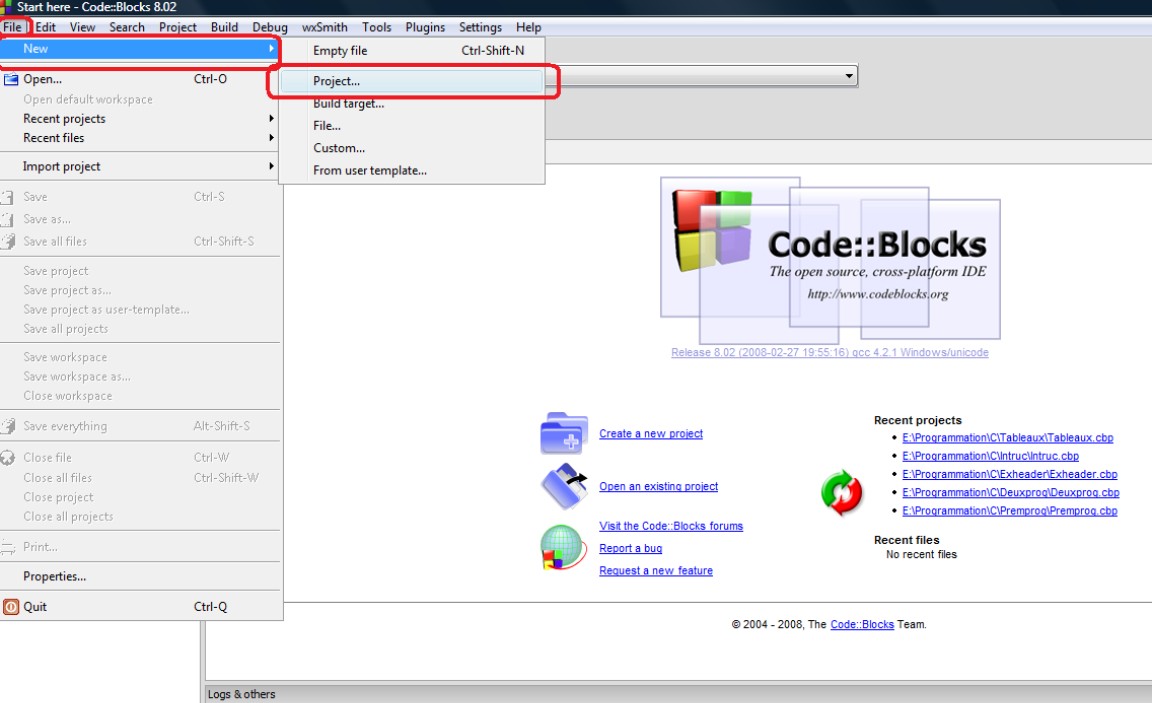
## Les outils de développement :

Les outils de développement jouent un rôle essentiel dans le processus de création, de test, de débogage et de déploiement de logiciels. Ils offrent des fonctionnalités et des ressources qui aident les développeurs à être plus efficaces et à produire un code de meilleure qualité. Voici quelques-unes des utilités principales des outils de développement :

* Productivité accrue : Les outils de développement fournissent des fonctionnalités telles que l'autocomplétion du code, la mise en surbrillance de la syntaxe, la navigation dans le code, le refactoring automatique, etc.
* Débogage facilité : Les outils de développement proposent des débogueurs qui aident les développeurs à identifier et à corriger les erreurs dans leur code.
* Collaboration simplifiée : Les outils de développement fournissent des fonctionnalités de gestion de versions (par exemple, Git) qui permettent aux développeurs de travailler en équipe et de collaborer sur un même projet.
* Tests automatisés : Les outils de développement proposent des Frameworks et des bibliothèques pour automatiser les tests unitaires, les tests fonctionnels et les tests de performance.
  1. Logiciel et outils utilisés :



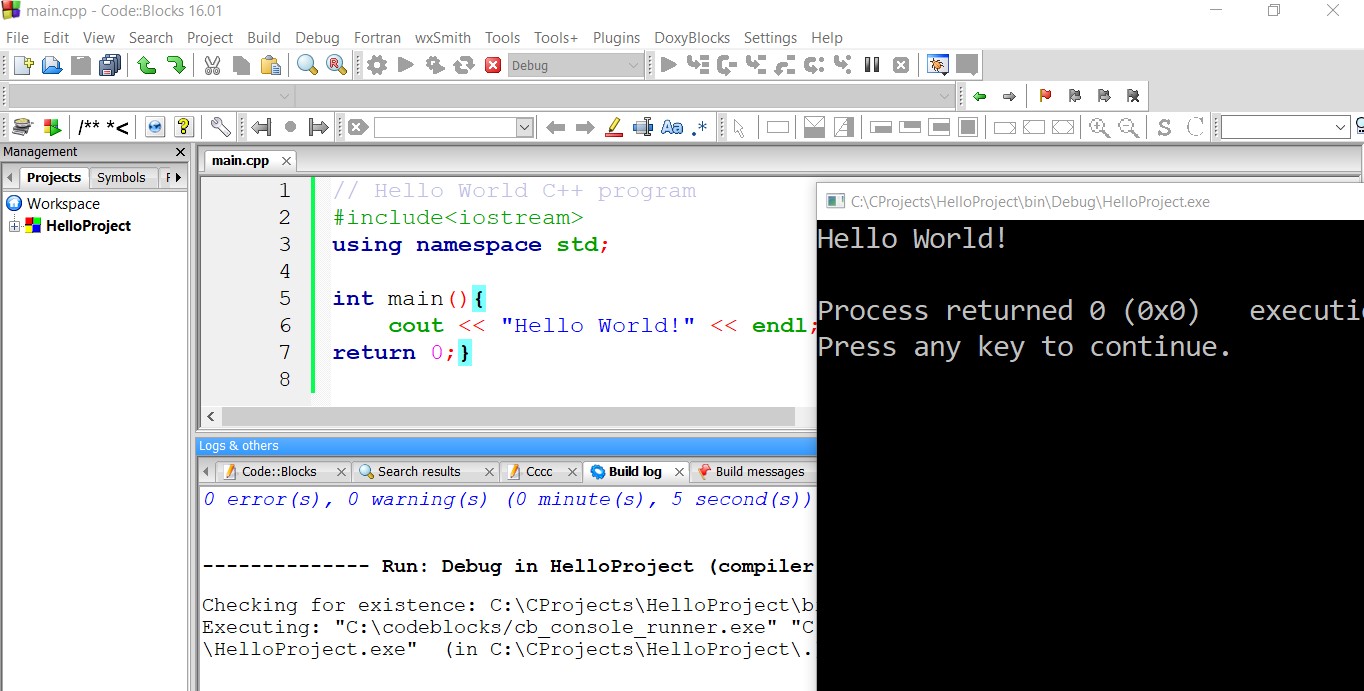
Code Blocks est une excellente option pour la programmation en C. Il s'agit d'un environnement de développement multiplateforme open source intégré qui supporte l'utilisation de compilateurs multiples comme : GCC (MingW/GNU GCC), MSVC , Digital Mars, Borland C 5.5 et Open Watcom. Le compilateur par défaut inclus avec cet ensemble Code Blocks est MinGW.



La fonctionnalité de Code Blocks peut être étendue par l'utilisation de plugiciels. Il possède est un système de construction rapide et personnalisé qui ne nécessite pas de makefiles, et vous permet d'effectuer des constructions parallèles sur des systèmes avec un microprocesseur multi-coeur.

Le débogueur de Code Block supporte l'application de points d'arrêt sur le code source ou sur les données que le programme manipule, ainsi que l'établissement de conditions et compteurs pour lesdits points d'arrêt. Vous pouvez aussi créer des inspections définies par l'utilisateur. Une autre fonctionnalité digne de mention est les vidages de mémoire personnalisés.

Dernièrement, Code Blocks ajoute de la couleur à la syntaxe du code source, inclut une fonction de pliage de code pour les langues C et XML, une interface par onglets, un navigateur de classe et une fonctionnalité pour la complétion du code source.



## Langages utilisés

### Formation Langage C - Développez avec le langage C - SparksLe C est un langage compilé (par opposition aux langages interprétés). Cela signifie qu'un programme C est décrit par un fichier texte, appelé fichier source. Ce fichier n'étant évidemment pas exécutable par le microprocesseur, il faut le traduire en langage machine.

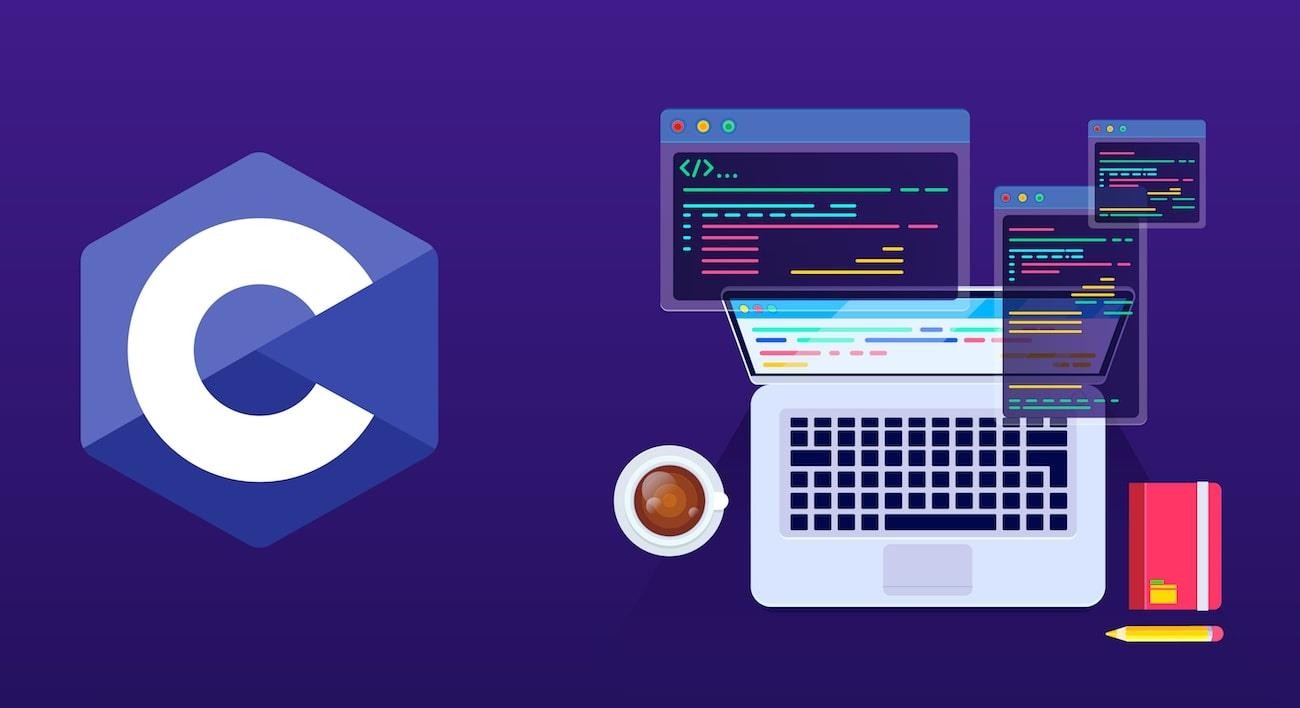
Cette opération est effectuée par un programme appelé compilateur. La compilation se décompose en fait en 4 phases successives :

* + - *Le traitement par le préprocesseur :* le fichier source est analysé par le préprocesseur qui effectue des transformations purement textuelles (remplacement de chaînes de caractères, inclusion d'autres fichiers source

...).

* + - *La compilation :* la compilation proprement dite traduit le fichier généré par le préprocesseur en assembleur, c'est-à-dire en une suite d'instructions du microprocesseur qui utilisent des mnémoniques rendant la lecture possible.
    - *L'assemblage :*cette opération transforme le code assembleur en un fichier binaire, c'est-à-dire en instructions directement compréhensibles par le processeur. Généralement, la compilation et l'assemblage se font dans la foulée, sauf si l'on spécifie explicitement que l'on veut le code assembleur. Le fichier produit par l'assemblage est appelé fichier objet.
    - *L'édition de liens :* un programme est souvent séparé en plusieurs fichiers source, pour des raisons de clarté mais aussi parce qu'il fait généralement appel à des librairies de fonctions standard déjà écrites. Une fois chaque code source assemblé, il faut donc lier entre eux les différents fichiers objets. L'édition de liens produit alors un fichier dit exécutable.

Les différents types de fichiers utilisés lors de la compilation sont distingués par leur suffixe. Les fichiers source sont suffixés par .c, les fichiers prétraités par le préprocesseur par .i, les fichiers assembleur par .s, et les fichiers objet par .o. Les fichiers objets correspondant aux librairies pré-compilées ont pour suffixe .a.



## Les structures utilisées :

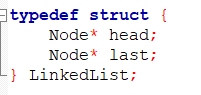
La structure Node représente un nœud dans une liste chaînée doublement liée. Chaque nœud contient :

* data : un caractère qui stocke la valeur du nœud (utilisé pour stocker les symboles 'X', 'O' ou un espace vide).
* next : un pointeur vers le nœud suivant dans la liste.
* prev : un pointeur vers le nœud précédent dans la liste.

## 

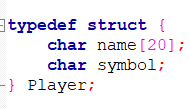
La structure LinkedList représente une liste chaînée doublement liée. Elle contient :

* head : un pointeur vers le premier nœud de la liste.
* last : un pointeur vers le dernier nœud de la liste.



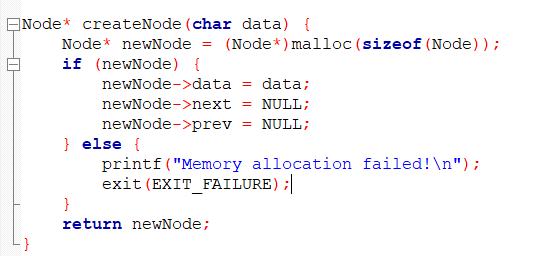
La structure Player représente un joueur dans le jeu de Tic Tac Toe. Elle contient :

* name : un tableau de caractères pour stocker le nom du joueur (jusqu'à 19 caractères plus le caractère nul de fin de chaîne).
* symbol : un caractère pour stocker le symbole du joueur ('X' ou 'O').

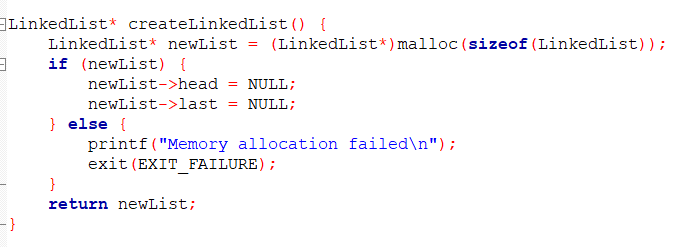


## Les fonctions utilisées :

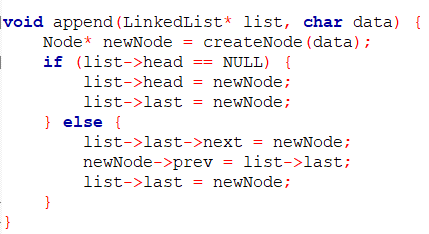
* + - La fonction createNode est chargée de créer un nouveau nœud pour une liste chaînée. Elle prend en paramètre une donnée de type caractère et crée dynamiquement un nouveau nœud en allouant de la mémoire pour celui-ci. Ensuite, elle initialise les champs du nœud avec la donnée spécifiée et les pointeurs next et prev à NULL. Si l'allocation de mémoire échoue, elle affiche un message d'erreur et quitte le programme avec un code d'échec



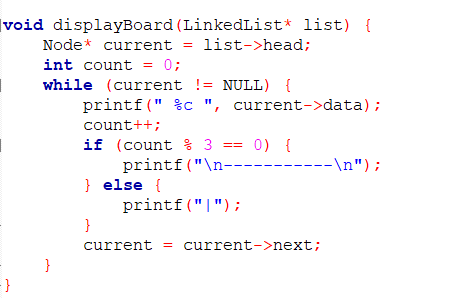
* + - La fonction createLinkedList est responsable de la création d'une nouvelle liste chaînée vide. Elle alloue dynamiquement de la mémoire pour la structure de liste chaînée, puis initialise les pointeurs de tête et de fin de la liste à NULL. Si l'allocation de mémoire échoue, elle affiche un message d'erreur et quitte le programme avec un code d'échec.

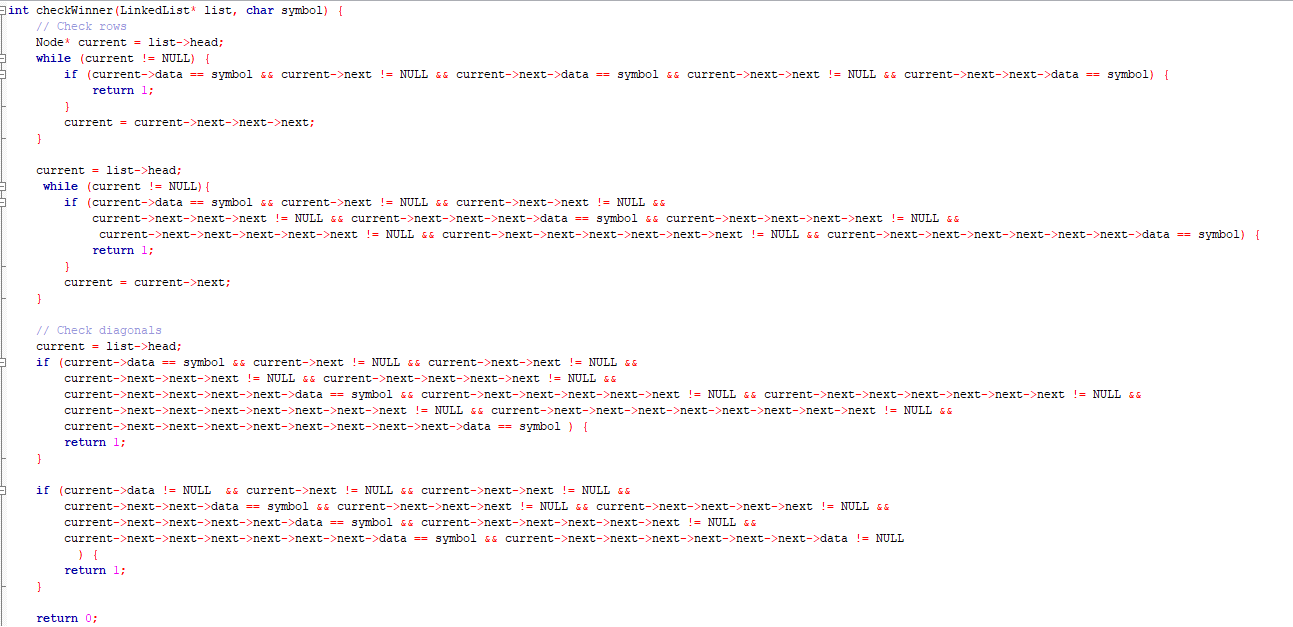
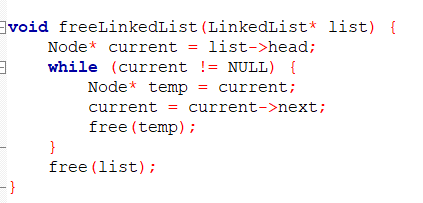


* + - La fonction append ajoute un nouveau n avec une donnée spécifiée à la fin de la liste chaînée donnée. Si la liste est vide, elle crée le premier nœud avec la donnée. Sinon, elle parcourt la liste jusqu'à atteindre le dernier nœud et ajoute le nouveau nœud après celui-ci. Elle met à jour le pointeur de fin de la liste pour pointer vers le nouveau nœud ajouté.



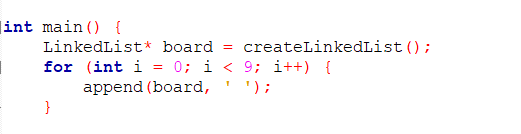
* + - La fonction displayBoard affiche l'état actuel du jeu de tic-tac-toe représenté par la liste chaînée. Elle parcourt la liste chaînée en imprimant les caractères de chaque nœud, ce qui correspond aux cellules de la grille de jeu. Elle ajoute également des séparateurs pour représenter les lignes et les colonnes de la grille.



* + -  La fonction checkWinner vérifie s'il y a un gagnant dans le jeu de tic-tac-toe en analysant la grille de jeu représentée par la liste chaînée. Elle prend en paramètres la liste chaînée et le symbole (X ou O) à vérifier. Elle examine les lignes, les colonnes et les diagonales de la grille pour voir si le symbole spécifié apparaît trois fois de manière consécutive, ce qui indiquerait un joueur gagnant.
    -  La fonction freeLinkedList est chargée de libérer la mémoire allouée dynamiquement pour tous les nœuds de la liste chaînée, ainsi que pour la structure de liste elle-même. Elle parcourt la liste chaînée et libère chaque nœud un par un en utilisant la fonction free, puis libère la mémoire de la structure de liste chaînée elle-même

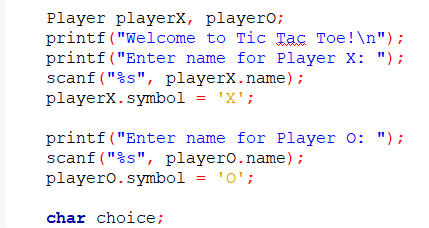
## Fonction main :

## Initialisation de la grille de jeu:



**Initialisation de la grille de jeu :** Une nouvelle liste chaînée, board, est créée pour représenter la grille de jeu de Tic Tac Toe. La grille est initialement remplie de 9 espaces vides (' '), représentant les 9 cases vides.

## Initialisation de la grille de jeu:



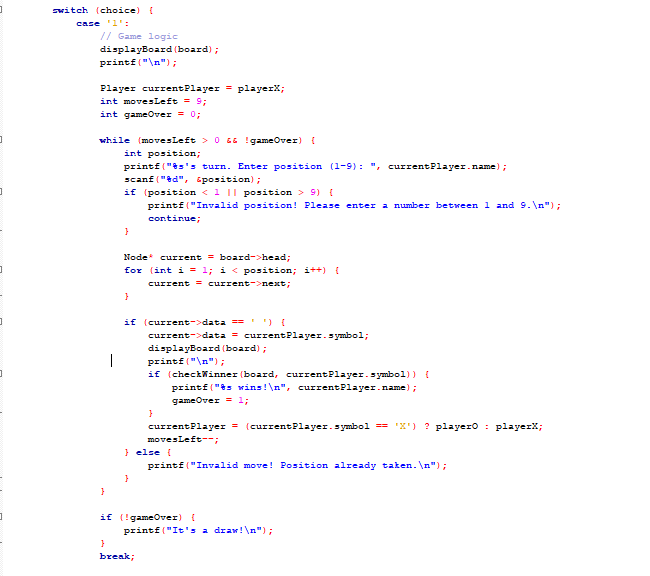
**Les joueurs playerX et playerO sont initialisés. Le programme demande aux utilisateurs de saisir les noms des deux joueurs. Les symboles 'X' et 'O' sont attribués respectivement aux joueurs playerX et playerO.**

## Boucle du menu principal:

## 

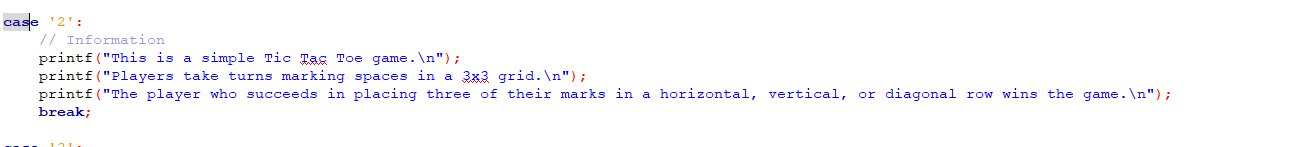
## **Un menu est affiché pour que l'utilisateur puisse choisir entre jouer au jeu, obtenir des informations sur le jeu, ou quitter.**

## Gestion des choix du menu:



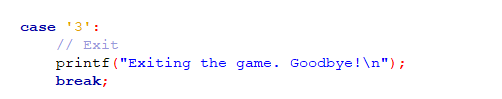
Lorsqu'un utilisateur choisit de jouer (case '1'), la grille de jeu est affichée, et la boucle de jeu commence. La grille de jeu est d'abord affichée pour montrer l'état initial. Les variables de jeu sont initialisées avec currentPlayer assigné à playerX, movesLeft à 9, et gameOver à 0. La boucle de jeu continue tant qu'il y a des coups disponibles (movesLeft > 0) et que le jeu n'est pas terminé (!gameOver). Durant le tour du joueur actuel (currentPlayer), celui-ci entre la position de son coup. La position est validée pour s'assurer qu'elle est comprise entre 1 et 9. Si elle est invalide, un message d'erreur est affiché et le tour recommence. Si la position est valide et libre, la grille est mise à jour avec le symbole du joueur ; sinon, un message d'erreur est affiché. Après chaque coup, la fonction checkWinner est appelée pour vérifier si le joueur actuel a gagné. En cas de victoire, un message est affiché, gameOver est mis à 1, et la boucle se termine. Si aucun joueur n'a gagné, le joueur courant change (de playerX à playerO ou vice versa), et le nombre de coups restants (movesLeft) est décrémenté. Si aucun joueur n'a gagné après tous les coups, un message de match nul est affiché

Choix des informations



**Si l'utilisateur choisit l'option d'information (case '2'), une brève description du jeu de Tic Tac Toe est affichée**

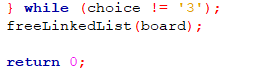
Choix de quitter

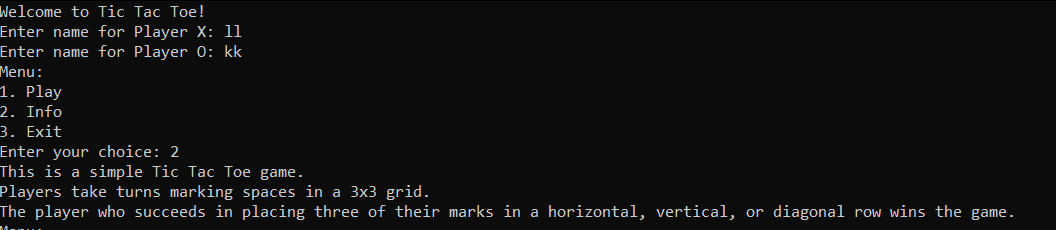
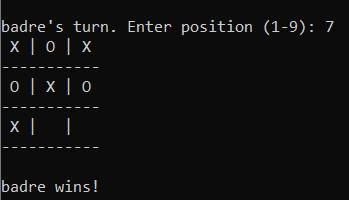
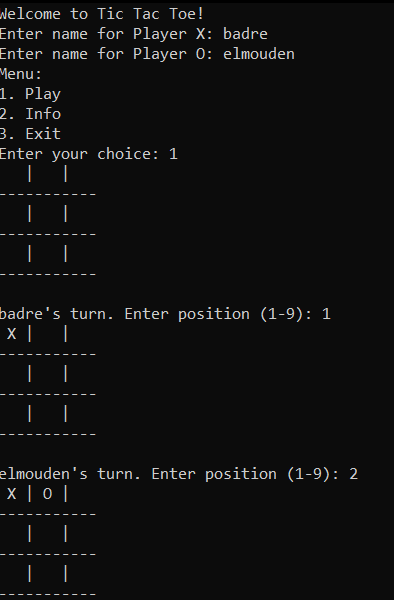


Choix invalide



Libération de la mémoire



 Les resultats:

*Conclusion :*

Ce projet de jeu Tic Tac Toe en langage C a couvert plusieurs aspects clés de la programmation, en mettant l'accent sur la création et la manipulation de structures de données dynamiques, ainsi que sur l'interaction utilisateur. Nous avons développé un système robuste et fonctionnel qui permet aux utilisateurs de jouer à un jeu de Tic Tac Toe de manière intuitive et fluide.

Notre système de jeu Tic Tac Toe en langage C offre une solution complète et fonctionnelle pour gérer les parties, vérifier les conditions de victoire, et assurer une interaction conviviale avec les joueurs. Il peut être facilement étendu et amélioré pour ajouter des fonctionnalités supplémentaires ou pour répondre à des exigences spécifiques.

En conclusion, ce projet démontre notre capacité à concevoir, développer et mettre en œuvre un jeu de Tic Tac Toe robuste en langage C. Nous espérons que notre travail contribuera à l'apprentissage et à la satisfaction des utilisateurs dans le domaine de la programmation de jeux.

# Sources :

*Diagramme de classe :*

<https://www.ibm.com/docs/fr/rsar/9.5?topic=diagrams-class>

*L'UML :*

<https://www.futura-sciences.com/tech/definitions/informatique-uml-3979/>

*Les outils de développement :*

<https://www.startechup.com/fr/blog/software-development-technologies/>

*Code-blocks :*

<https://code-blocks.fr.uptodown.com/windows>

*Langage c :*

<https://www.rocq.inria.fr/secret/Anne.Canteaut/COURS_C/chapitre1.html>